Dharmsinh Desai University, Nadiad

Faculty of Technology

Department of Computer Engineering

ASSIGNMENT 7

Subject: Smart Device Programming

Submitted To:

Prof. Jignesh Shah

CE Department

Submitted By:

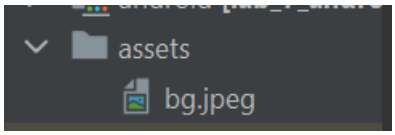
Sherathiya Dhruvi A.

Student ID:20CEUOS006

Roll No.: CE126

Adding Images in Flutter

To add image in Flutter app, first of all, create an assets folder then add the actual images inside the folder. After this, Add the image path in pubspec.yaml and then display it using the image: AssetImage() widget.



First we need to set out assets path in pubspec.yaml file. So will do that as shown below:

Now create one stateful widget named FirstClass. For this write stf and then press tab key. This newly created widget will return Scaffold which contains appBar, body and floatingActionButton.

Scaffold: The Scaffold is **a widget in Flutter used to implements the basic material design visual layout structure**. It is quick enough to create a general-purpose mobile application and contains almost everything we need to create a functional and responsive Flutter apps. This widget is able to occupy the whole device screen.

appBar: AppBar is **usually the topmost component of the app (or sometimes the bottom-most), it contains the toolbar and some other common action buttons**.

body: The widget in the body of the scaffold is positioned at the top-left of the available space between the app bar and the bottom of the scaffold. To center this widget instead, consider putting it in a Center widget and having that be the body.

floatingActionButton: Use at most a single floating action button per screen. Floating action buttons should be used for positive actions such as "create", "share", or "navigate".

```dart
class _FirstClassState extends State<FirstClass> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Flutter Lab 7'),
        centerTitle: true,
        backgroundColor: Color(0xff22223b),
      ),  // AppBar
      body: Center(
          child: Image(
              image: AssetImage('assets/bg.jpeg'),
          )  // Image
      ),  // Center
      floatingActionButton: FloatingActionButton(
          onPressed: () {},
          child: Text('Click'),
          backgroundColor: Color(0xff22223b),
      ),  // FloatingActionButton
    );  // Scaffold

  }
}
```

Adding Icons in Flutter

Flutter provides an **Icon Widget** to create icons in our applications. We can create icons in Flutter, either using inbuilt icons or with the custom icons. Flutter provides the list of all icons in the **Icons class**.

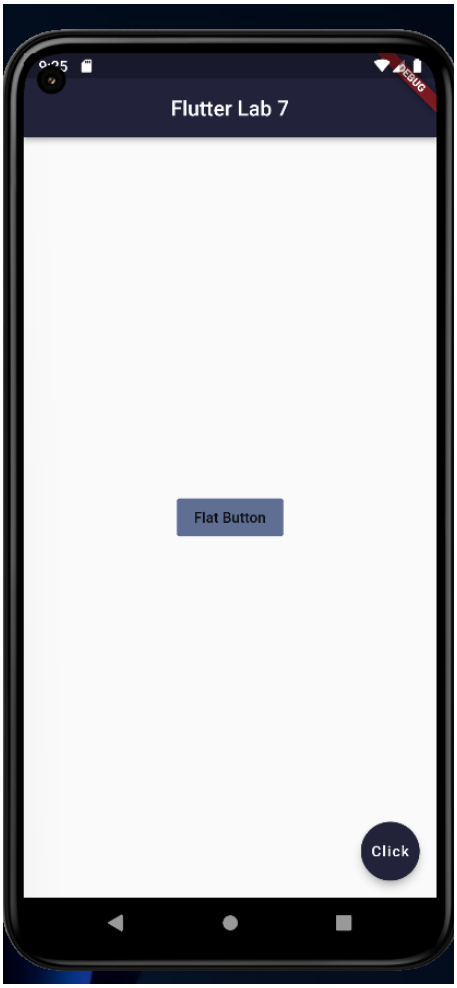
```
class _SecondClassState extends State<SecondClass> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Flutter Lab 7'),
        centerTitle: true,
        backgroundColor: Color(0xff22223b),
      ),  // AppBar
      body: Center(
          child: Icon(
              Icons.fingerprint,
              color: Colors.black,
              size: 80.0,
            )  // Icon
      ),  // Center
      floatingActionButton: FloatingActionButton(
        onPressed: () {},
        child: Text('Click'),
        backgroundColor: Color(0xff22223b),
      ),  // FloatingActionButton
    );  // Scaffold
  }
}
```

Here we are using icons with Icon widget and inside that we have defined icon name, icon color and its size.

Adding Buttons in Flutter

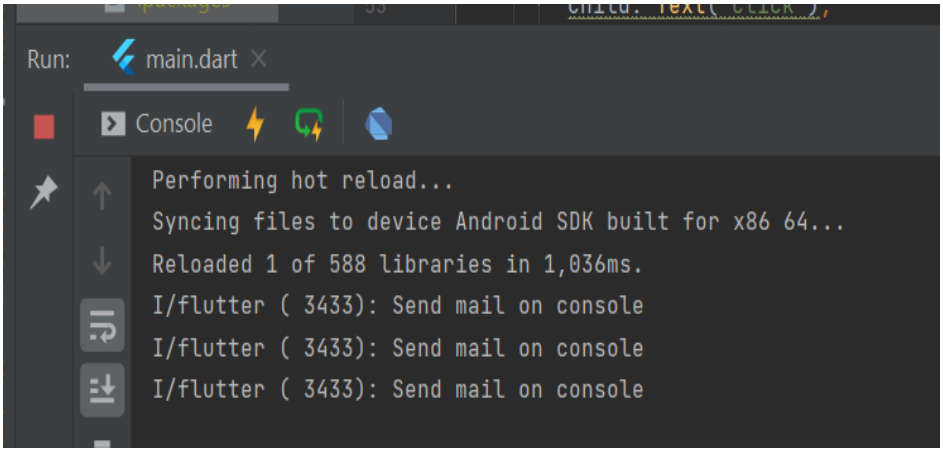
Buttons are the Flutter widgets, which is a part of the material design library. Flutter provides several types of buttons that have different shapes, styles, and features. There are many types of buttons in flutter like ElevatedButton, FlatButton , IconButton and TextButton.

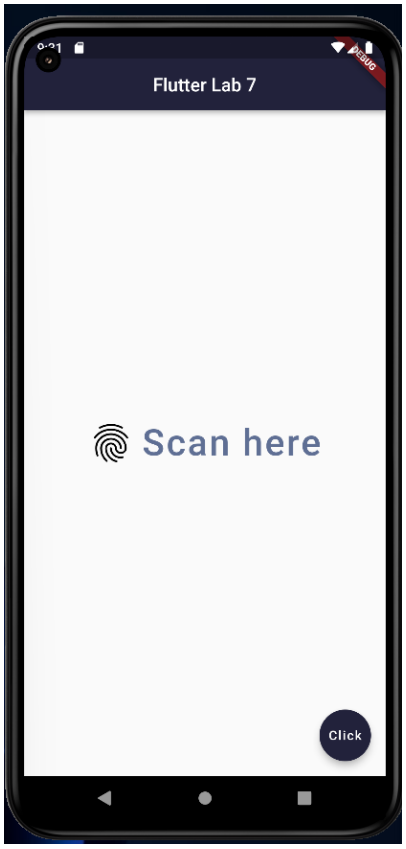
```
body: Center(
  child: ElevatedButton(
    child: Text('Button'),
    onPressed: () {},
    style: ElevatedButton.styleFrom(
        primary: Color(0xff22223b),
        padding: EdgeInsets.symmetric(horizontal: 50, vertical: 20),
        textStyle: TextStyle(
            fontSize: 40,
            fontWeight: FontWeight.bold)),  // TextStyle

    ),  // ElevatedButton
),  // Center
```




```
body: Center(
  child: FlatButton(
    onPressed: () { print('print on consol'); },
    child: Text('Flat Button'),
    color: Color(0xff5F6F94),
  ),  // FlatButton
),  // Center
```

```dart
body: Center(
  child: IconButton(
    icon: Icon(
      Icons.mail_outline_sharp,
      size: 30.0,
    ),  // Icon
    tooltip: 'send mail me',
    onPressed: () {
      print('Send mail on console');
    },
  ),  // IconButton
),  // Center
```

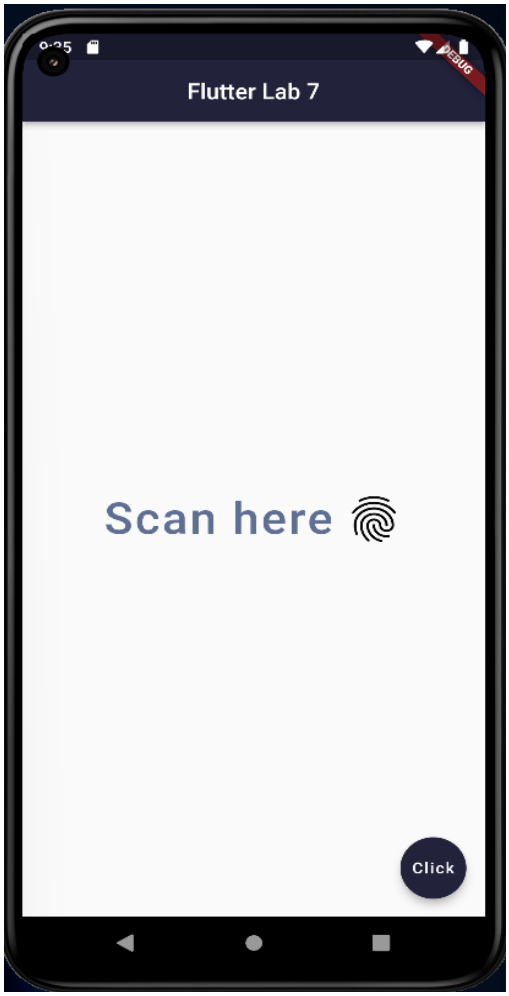





```dart
body: Center(
  child: TextButton.icon(
    icon: Icon(
      Icons.fingerprint,
      color:Colors.black,
      size: 50.0,
    ),  // Icon
    label: Text(
      "Scan here",
      style: TextStyle(
        color: Color(0xff5F6F94),
        fontSize: 40.0,
        letterSpacing: 2.0,
      ),  // TextStyle
      textAlign: TextAlign.start,
    ),  // Text
    onPressed: () {},
  ),  // TextButton.icon
),  // Center
```

We can change icon and it’s text direction by using textDirection property described below:

```
body: Center(
  child: Directionality(
    textDirection: TextDirection.rtl,
    child: TextButton.icon(
      icon: Icon(
        Icons.fingerprint,
        color:Colors.black,
        size: 50.0,
      ),  // Icon
      label: Text(
        "Scan here",
        style: TextStyle(
          color: Color(0xff5F6F94),
          fontSize: 40.0,
          letterSpacing: 2.0,
        ),  // TextStyle
        textAlign: TextAlign.start,
      ),  // Text
      onPressed: () {},
    )  // TextButton.icon
  )  // Directionality
),  // Center
```



GitHub Repository Link:

https://github.com/DhruviSherathiya/SmartDeviceProgramming/tree/main/Lab7